EP 25100 (1)

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04L 12/58, 9/30, 9/08, 29/06 — A1

(11) International Publication Number: **WO 00/42748**

(43) International Publication Date: 20 July 2000 (20.07.00)

(21) International Application Number: PCT/US00/01011

(22) International Filing Date: 14 January 2000 (14.01.00)

(30) Priority Data:
60/116,053 14 January 1999 (14.01.99) US

(71) Applicant: TUMBLEWEED COMMUNICATIONS CORP. [US/US]; 700 Saginaw Drive, Redwood City, CA 94063 (US).

(72) Inventors: DOLINSKY, Dmitry; 700 Saginaw Drive, Redwood City, CA 94063 (US). BANDINI, Jean-Christophe; 10230 N. Foothill Boulevard, #E10, Cupertino, CA 95014 (US).

(74) Agent: IVEY, James, D.; Law Offices of James D. Ivey, 3025 Totterdell Street, Oakland, CA 94611-1742 (US).

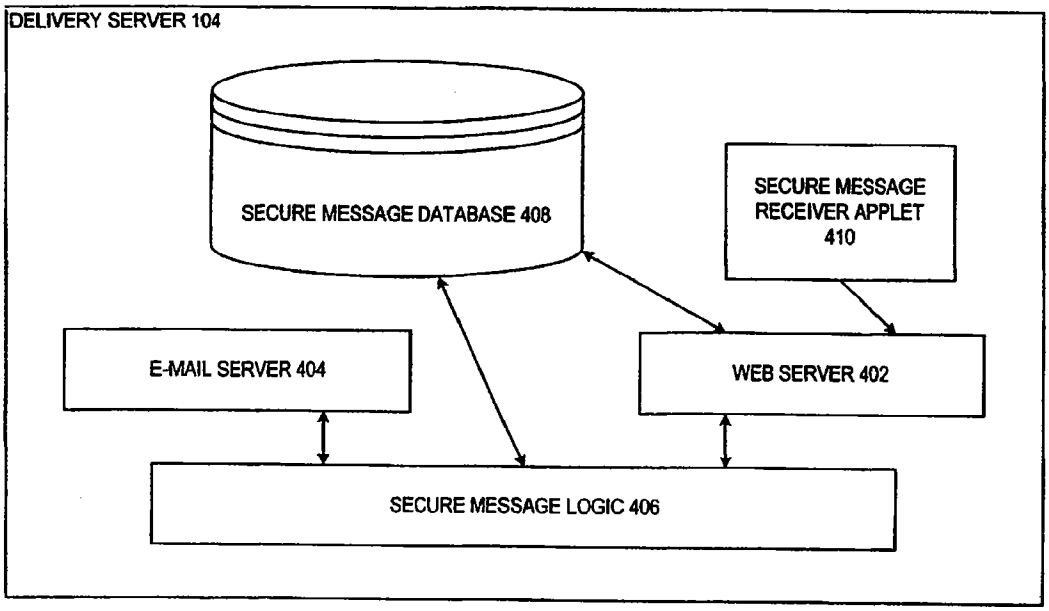
(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: WEB-BASED DELIVERY OF SECURE E-MAIL MESSAGES



(57) Abstract

An electronic document delivery system that enables exchange of digitally signed and public key encrypted packages without requiring pre-installed specialized public key security software applications on recipient's computer. A URL identifying such a package is sent to the recipient by e-mail. Upon submission of the URL through the World Wide Web, computer instructions in the form of an applet, for example, are sent to the recipient to effect retrieval and processing of the package. Such processing can include decrypting one or more parts of the package and verification of a signature of the package.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

WEB-BASED DELIVERY OF SECURE E-MAIL MESSAGES

SPECIFICATION

FIELD OF THE INVENTION

The invention relates generally to the secure delivery of data files through a computer network and, in particular, to delivery of data files which are cryptographically encrypted and/or signed in such a manner that the data files can be decrypted and the signatures can be verified.

BACKGROUND OF THE INVENTION

As electronic mail (e-mail) has become a widely available and widely used form of communication, security of such e-mail communication has become an important issue. An important development in secure e-mail communication is the S/MIME (Secure Multipurpose Internet Mail Extension) protocol. S/MIME is a security protocol designed by RSA Data Security, Inc. and other software companies in 1995 to provide message integrity, authentication, non-repudiation and privacy to electronic message exchange through the use of digital signatures and encryption.

S/MIME relies on MIME (Multipurpose Internet Mail Extensions) for specifying content and structure of S/MIME messages and PKCS (Public-Key Cryptography Standards) for cryptographic functionality. Certificates conforming to the known ITU-T X.509 standard, i.e., X.509 certificates, are used to identify communicating parties.

Currently, there are three versions of S/MIME standard. The most popular is version 2. Version 3 of S/MIME became a standard in July 1999 and introduces a number of enhancements to S/MIME. S/MIME version 3 also introduces a new Cryptographic Message Syntax that is an extension and replacement of PKCS #7 v1.5 used in S/MIME version 2.

More information regarding S/MIME and RSA Data Security, Inc. can be found

on the RSA Data Security, Inc. web site (http://www.rsa.com). The current status of S/MIME standardization and other S/MIME related documents can be found on IMC (Internet Mail Consortium) web site (http:///www.imc.org) and IETF S/MIME Working group web sites (http://www.imc.org/ietf-smime/, http://www.ietf.org/html.charters/smime-charter.html).

The S/MIME version 2 standard is published in two RFC documents:

- S/MIME Version 2 Message Specification (RFC 2311)
- S/MIME Version 2 Certificate Handling (RFC 2312)

S/MIME Version 3 is currently specified in the following collection of the Internet Draft documents:

- Cryptographic Message Syntax (draft-ietf-smime-cms)
- S/MIME Version 3 Message Specification (draft-ietf-smime-msg)
- S/MIME Version 3 Certificate Handling (draft-ietf-smime-cert)
- Certificate Request Syntax (draft-ietf-smime-crs)
- Enhanced Security Services for S/MIME (draft-ietf-ietf-ess)
- Signing Certificate Attribute Specification (draft-ietf-smime-sigattr)
- Certificate Distribution Specification (draft-ietf-smime-certdist)
- Diffie-Hellman Key Agreement Method (draft-ietf-smime-x942)
- Domain Security Services using S/MIME (draft-ietf-smime-domsec)

These documents are available from the web sites listed above or from many other web sites such as the web site for RFC Editor (http://www.rfc-editor.org).

The current practice of exchanging S/MIME messages calls for preparing an S/MIME file according to the S/MIME specifications and sending the S/MIME file as an attachment to an e-mail message. When the message reaches the recipient, it is expected that the recipient has an S/MIME-capable e-mail reader to process the message. Alternatively, the recipient can use a stand alone S/MIME processing helper application to process the message.

There are many obstacles to successful exchange of the S/MIME messages

using the scheme described above. First, the recipient may simply lack S/MIME capability such that the S/MIME message is unusable and is simply stored in the recipient's computer without further handling. There are many circumstances that may lead to this situation. The recipient's e-mail reading program of choice may be a simple e-mail reader or an early version without the ability to properly authenticate and decrypt messages in S/MIME format. Alternatively, the e-mail reader of the recipient can rely on plug-ins or helper programs that are not installed on the recipient computer. In any case, a message in S/MIME format is just unreadable to the recipient and the recipient would probably not understand why this particular email message is unreadable without additional communication with the sender.

S/MIME messaging generally requires that both the sender and recipient are aware of infrastructure required for public key cryptography, i.e., public key infrastructure (PKI). Such infrastructure includes public/private key pairs required for encryption, decryption, and signing and certificates required for authentication. The recipient may be unaware of such public key infrastructure in general and S/MIME in particular. However, to receive encrypted secure e-mail, the recipient must have a digital certificate which includes the pubic key of the recipient. If the recipient has a certificate, some familiarity with public key infrastructure by the recipient can be assumed. However, in case of signed message no public or private key of the recipient is required and, accordingly, no such assumption can be made with respect to the recipient's familiarity with public key infrastructure.

A second obstacle to successful delivery and processing of a message in the S/MIME format is inter-version incompatibility. Even if the recipient's e-mail reader does have some level of S/MIME capability, S/MIME processing may fail due to a difference in S/MIME implementation by different vendors. S/MIME is an evolving standard; there are three different versions in use currently. If the sender uses an S/MIME program implementing a later version than the one implemented by the recipient's e-mail program, the recipient's e-mail reader could be unable to properly decrypt the message and verify the signature. As in the case described above, the message may simply appear to be unreadable and the recipient may not be able to

determine the course of action needed to remedy the situation.

The ordinary solution for such problems is to have recipients upgrade or obtain their e-mail reader or helper applications to properly implement the most recent standardized version of the S/MIME protocol. Installing additional software to a computer system, even simply to upgrade an existing application, can have adverse effects on the computer system by occupying additional system resources. Accordingly, the recipient can choose not to install new S/MIME software. In addition, the appropriate S/MIME software may not be available for the recipient's computing environment. Even if the recipient chooses to upgrade or obtain the S/MIME software, such installation can take considerable time and significantly delay viewing of the message by the recipient.

S/MIME e-mail messages have another shortcoming. S/MIME messages are transferred as attachments to Internet e-mail. Some Internet e-mail gateways limit the size of the message and therefore the size of such attachments. Accordingly, the S/MIME message can be truncated. If the message is truncated, a portion of the data needed to properly decrypt the message and verify the signature is lost and the message becomes useless.

What is needed is a mechanism by which e-mail can be sent to a recipient using a standardized, wide-supported secure e-mail protocol and by which the recipient can receive the e-mail notwithstanding failure of the recipient's e-mail reader to properly implement the secure e-mail protocol.

SUMMARY OF THE INVENTION

In accordance with the present invention, messages which comport with a secure e-mail standard, such as S/MIME, are delivered through a document delivery protocol such as the HTTP protocol of the World Wide Web rather than a conventional e-mail protocol such as SMTP/POP. Web-based secure e-mail delivery increases the likelihood of successful delivery by removing the burden of obtaining and maintaining secure e-mail processing software from the recipient and by having the recipient download the secure message from a Web server instead of receiving it as Internet mail attachment.

The secure e-mail delivery system uses a notification message sent to the recipient by regular e-mail and a web server for delivery of the secure message through the World Wide Web according to HTTP. As a result, secure e-mail can be sent to virtually any recipient with an e-mail account, regardless of the capabilities of the recipient's computer system beyond a commonly available web browser and e-mail reader.

The secure e-mail delivery system exploits the ability of commonly available web browsers to automatically install and run software from a web server. Particularly useful is ability of such a browser to invoke Java™ applets located on the server such that the software can be brought to the widest range of computing environments. Some web browsers currently have the ability to automatically download and install other types of software including plug-in applications and known ActiveX controls. In particular, the recipient responds to the notification message by submitting a URL through the World Wide Web to a web server which controls access to the secure e-mail message. In response, the web server sends, in addition to the secure e-mail message itself, software which processes the secure message to thereby render the message readable to the recipient. Such processing includes, for example, decrypting the message and one or more attached data files and verifying a signature of the secure e-mail message.

There are two main components to the secure e-mail delivery system. The first component is the web server that accepts secure messages for delivery. The second component is the secure message processing software. The sender is assumed to have software capable of generating secure e-mail messages according to a secure e-mail message standard such as S/MIME, PGP/MIME, or OpenPGP. It is also assumed that the recipient has access to Internet e-mail with the ability to read short text messages — attachment capabilities are not required — and has a web browser capable of running an applet, and/or downloading and installing software, such as the secure message processing software.

To send the secure message, the sender generates a secure message and submits the secure message to the web server of the secure e-mail delivery system along with the e-mail address of the intended recipient. Such submission can happen in many different ways: for example, through a web form hosted by the web server, using secure e-mail authoring software which is aware of the secure e-mail delivery system and the web server,

or through the use of an Internet e-mail relay which is aware of the secure e-mail delivery system and the web server. Specifically, the e-mail relay can detect e-mail which comports with a secure e-mail protocol supported by the secure e-mail delivery system and redirect such e-mail to the secure e-mail delivery system and allow other e-mail to continue transit according to a conventional e-mail protocol such as SMTP.

Upon such submission, the secure e-mail delivery system generates an e-mail message to the recipient informing the recipient about the secure message and including the Uniform Resource Locator (URL) to be used to retrieve the secure message. This URL points to the web server of the secure e-mail delivery system and contains enough information so that web server can locate the secure message.

The recipient uses the web browser to access the URL on the web server. The web server generates the web page that instructs the web browser to invoke secure e-mail processing software. The recipient's web browser causes execution of the secure e-mail processing software which retrieves the secure message from the web server, processes the secure message, and presents the result to the recipient. Processing of the secure message can include decrypting the message and one or more attached data files and verifying any signature of the message.

The secure e-mail delivery system according to the present invention enables the recipient to properly process secure messages without having installed secure message processing software prior to receiving the message. In addition, since the secure message processing software is stored on the web server, the secure e-mail delivery system can ensure that the recipient has the latest available version of the secure message processing software.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram which depicts a secure e-mail delivery system including a delivery server which delivers secure e-mail in accordance with the present invention.

Figure 2 is a block diagram showing the sender of Figure 1 in greater detail.

Figure 3 is a logic flow diagram illustrating generation of a secure e-mail message

according to one secure e-mail protocol.

Figure 4 is a block diagram of the delivery server of Figure 1 in greater detail.

Figures 5 and 6 are logic flow diagrams of delivery of a secure e-mail message according to the present invention.

Figure 7 is a block diagram of the recipient of Figure 1 in greater detail.

Figure 8 is a block diagram showing the secure e-mail receiver of Figure 7 in greater detail.

Figure 9 is a logic flow diagram of the receipt of a secure e-mail message in accordance with the present invention.

Figure 10 is a logic flow diagram of the processing of a secure e-mail message in accordance with one secure e-mail protocol.

DETAILED DESCRIPTION

In accordance with the present invention, a delivery server 104 (Figure 1) effects delivery of an e-mail message according to a secure e-mail protocol by (i) notifying a recipient 106 of the e-mail message, (ii) receiving data from the recipient identifying the message, and (iii) supplying, in response to the data, computer instructions which effect receipt and proper handling of the e-mail message according to the secure protocol.

In particular, a sender 102 uses a secure e-mail protocol to send a secure e-mail message through computer network 108, which can be the Internet for example. The secure e-mail message is addressed to recipient 106. However, in a manner described more completely below, the secure message is routed to delivery server 104. While delivery server 104 is shown as a single computer system in Figure 1, it is appreciated that delivery server 104 can include multiple computer systems.

Delivery server 104 assumes that recipient 106 does not have the appropriate software to properly handle the secure message according to the secure protocol. Accordingly, delivery server 104 assumes responsibility for proper implementation of the secure e-mail protocol within recipient 106. One such secure e-mail protocol is the S/MIME protocol, version 3. Other versions of the S/MIME protocol can also be the secure e-mail protocol. In alternative embodiments, other secure e-mail protocols such as

PGP/MIME and OpenPGP can be implemented.

As described more completely below, delivery server 104 sends a notification e-mail message using ordinary techniques to recipient 106. The notification e-mail message contains sufficient identification information to identify the specific secure message addressed to recipient 106 within delivery server 104.

When recipient 106 receives and reads the notification e-mail message, recipient 106 submits the identification information to delivery server 104 to thereby request delivery of the secure message. In this embodiment, the identification information is a universal resource locator (URL) which includes data identifying the secure message and recipient 106 sends the URL to delivery server 104 using the hypertext transfer protocol (HTTP), for example, through a web browser 704 (Figure 7) such as Netscape Navigator available from Netscape Corporation of Mountain View, California or Internet Explorer available from Microsoft Corporation of Redmond, Washington.

In response to the URL, delivery server 104 sends computer instructions which are to be executed by recipient 106 and which implement the secure e-mail protocol with which the secure message comports. As described below in greater detail, the computer instructions comport with the Java™ computer instruction language implemented by most currently available web browsers in an illustrative embodiment. Such web browsers, in response to receipt of such computer instructions, begin interpretation of such computer instructions, i.e., execution of the computer instructions as they arrive. In an alternative embodiment, the computer instructions define a computer program which is installed in the computer system of recipient 106 in persistent storage such that subsequent secure e-mail messages can be processed without requiring a second delivery of the computer instructions. In this illustrative embodiment, delivery server 104 verifies that the program installed for recipient 106 is the most current program available and re-delivers and re-installs the computer instructions if the program is not the most current.

As described more completely below, execution of the received computer instructions causes recipient 106 to retrieve the secure message from delivery server 104 and to process the retrieved secure e-mail message according to the secure e-mail protocol implemented by the received computer instructions.

Thus, recipient 106 can securely receive e-mail according to an established secure

e-mail protocol without first separately acquiring and properly installing software specifically designed to implement the secure e-mail protocol.

Sender 102

As described above, sender 102 prepares and sends a message according to a secure e-mail protocol such as S/MIME. In addition, the message is addressed to recipient 104. Sender 102 is shown to be a computer system. However, it is understood that e-mail addresses generally to not specify a computer system but instead specify a human user or, alternatively, a computer process. For simplicity, sender 102 is therefore a human user of the computer system or a computer process executing within the computer system. Similarly, recipient 106 is a human user of the computer system shown in the figures as recipient 106 or a computer process executing within the computer system.

To prepare the message according to the secure e-mail protocol, sender 102 includes an e-mail authoring process 202 (Figure 2) which in turn includes a secure protocol module 204. E-mail authoring process 202 can be commercially available S/MIME-capable e-mail software, i.e., software which has the ability to produce S/MIME message given an initial set of documents and one or more recipient e-mail addresses. An example of such software is the WorldSecure/Mail e-mail client available from Worldtalk Corporation of Santa Clara, California (http://www.worldtalk.com). While secure protocol module 204 is shown to be an integral component of e-mail authoring process 202, it is appreciated that secure protocol module 204 can be a plug-in or a helper application which adds functionality to e-mail authoring process 202.

Logic flow diagram 300 (Figure 3) illustrates one possible example of the packaging of a message by e-mail authoring process 202 and secure protocol module 204. Various secure e-mail protocols use various cryptographic protocols in addition to the sign-encrypt protocol illustrated in logic flow diagram 300. For example, some secure e-mail protocols do not permit signing and/or can support such cryptographic protocols as encrypt-sign and sign-encrypt-sign. In addition, logic flow diagram 300 is simplified for clarity and involves steps which are known and conventional such as key management, certificate look-up and retrieval, and certificate validations, for example.

In step 302, e-mail authoring process 202 (Figure 2) collects the information elements of the message, namely, one or more data files to include with the message and e-mail addresses of one or more recipients. In step 304 (Figure 3), secure protocol module 204 (Figure 2) packages the one or more data files into a multipart MIME entity. In test step 306 (Figure 3), secure protocol module 204 determines whether the secure e-mail message is to be signed as specified by sender 102 through e-mail authoring process 202. Sender 102 can specify whether the message is to be signed using conventional user interface techniques or, alternatively, can use logic to automatically determine whether signing the message is desired.

If the message is to be signed, processing transfers to step 308. Conversely, if the message is not to be signed, step 308 is skipped.

In step 308, secure protocol module 204 signs the multipart MIME entity to form a signed data package. In this embodiment, the signed data package conforms to the known PKCS #7 data format.

In test step 310, secure protocol module 204 determines whether the message is to be encrypted. Sender 102 can specify whether the message is to be encrypted using conventional user interface techniques or, alternatively, can use logic to automatically determine whether encrypting the message is desired.

If the message is to be encrypted, processing transfers to step 312. Conversely, if the message is not to be signed, steps 312-314 are skipped.

In step 312, secure protocol module 204 converts the signed data package to a new multipart MIME entity.

In step 314, secure protocol module 204 encrypts the new multipart MIME entity to form an encrypted data package which, in this illustrative embodiment, conforms to the known PKCS #7 data format.

In step 316, the data package is returned as the message to be sent. The data package is the signed data package of step 308 if the data is signed but not encrypted or is the encrypted data package of step 314 if the data is encrypted. Accordingly, the secure message can be signed and/or encrypted.

Message Submission Module 206

Message submission module 206 communicates with delivery server 104 to submit the secure message for delivery. In particular, message submission module 206 directs the secure e-mail message created by e-mail authoring process 202 and secure protocol module 204 to delivery server 104. In one embodiment, message submission module 206 is a component of e-mail authoring process 202 such that a human user of sender 102 uses a single, integrated user interface to compose and send a secure message to recipient 106. In this embodiment, message submission module 206 uses HTTP to communicate with delivery server 104 through network 108 and, in particular, uses an HTTP "Post" command to send the secure message. HTTP and the HTTP "Post" command are known and are not described further herein.

In a more simple embodiment, message submission module 206 is a web browser with the ability to upload data files by use of the HTTP "Post" command. Such would require that the message created through e-mail authoring process 202 and secure protocol module 204 be saved within sender 102 as a data file itself and separately transferred to delivery server 104 through message submission module 206.

In a third embodiment, sender 102 sends the message formed according to logic flow diagram 300 by ordinary SMTP (Simple Mail Transfer Protocol). In this embodiment, secure e-mail addressed to recipient 106 is directed by an e-mail relay to delivery server 104.

Such a relay can be implemented using the well-known "sendmail" program to invoke another well-known program, namely, "procmail," to process each incoming e-mail message according to a script. The script first checks each e-mail message for secure content and forwards the e-mail message to delivery server 104 if the e-mail message contains secure content or forwards the e-mail message to a conventional SMTP e-mail server otherwise.

For the relay to process each e-mail message for recipient 106, all e-mail for recipient 106 is routed through the relay. Such can be accomplished in a number of ways. First, sender 102 can specify that all outbound e-mail is to be routed through the e-mail relay. In particular, sender 102 can specify such an e-mail relay as the outgoing SMTP server since most currently available e-mail readers/composers allow a user to specify the

outgoing SMTP server. Accordingly, all e-mail sent by sender 102 according to a secure e-mail protocol is delivered through a secure e-mail delivery system such as that implemented by delivery server 104. Second, the relay can be a part of a component of the routing architecture of network 108 itself. For example, a large company, a university, or a commercial Internet service provider can install such an e-mail relay as an integral part of its e-mail routing. As a result, all e-mail directed to and/or from the large company, university, or commercial Internet service provider is processed by such an e-mail relay and the secure e-mail delivery system of delivery server 104 is available to all recipients reachable therethrough. Third, routing information can be included in the e-mail address of recipient 106 such that sender 102 directs e-mail messages to recipient 106 to pass through the e-mail relay.

Delivery Server 104

Delivery server 104 is shown in greater detail in Figure 4. Delivery server 104 includes a web server 402 which in turn includes a customized filter to intercept and redirect all HTTP requests. In particular, all HTTP requests which are unrelated to secure messages delivered by delivery server 104 are directed to another, conventional HTTP web server. As an alternative to redirecting all HTTP requests, web server 402 can process ordinary HTTP requests in a conventional manner and simultaneously filter requests involving URLs which refer to secure messages to be delivered in the manner described herein. Web server 402 can use standard programming techniques to select specific HTTP requests for special processing, including CGI, NSAPI, Active Server Page, and Servlet, for example. Web server 402 is the primary interface between delivery server 104 and recipient 106 in the delivery of the secure message created in accordance with a secure e-mail protocol such as that illustrated by logic flow diagram 300 as described above.

Delivery server 104 also includes an e-mail server 404. E-mail server 404 sends notification messages to an intended recipient in a manner described more completely below.

Delivery server 104 includes secure message logic 406 which governs the overall operation and interaction of the components of delivery server 104.

Once the secure message addressed to recipient 106 is received by delivery server

104, secure message logic 406 causes delivery of the secure message according to logic flow diagram 500 (Figure 5). In step 502, secure message logic 406 initiates delivery process with recipient 106 by generating a URL which contains key data to uniquely identify the secure message within a secure message database 408. In step 504, secure message logic 406 (Figure 4) sends a notification message including the URL and the key data contained therein to recipient 106 through e-mail server 404. The notification message is sent to recipient 106 using the recipient e-mail address included in the secure message and using the known SMTP protocol.

The general paradigm of delivering data by sending a URL which references the data to a recipient is described in U.S. Patent 5,970,970 to Smith et al. for "Electronic Document Delivery System in which Notification of Said Electronic Document is Sent to a Recipient Thereof" and that description is incorporated herein by reference. Inclusion of data identifying a message to be delivered is described in U.S. Patent Application S/N 08/832,784 by Jeffrey C. Smith and Jean-Christophe Bandini for "Private, Trackable URLs for Directed Document Delivery" and that description is incorporated herein by reference.

The remainder of the delivery of the secure message is accomplished through a "pull" paradigm according to logic flow 600 (Figure 6). In particular, recipient 106 (Figure 1) receives the notification message which includes the URL and the key data contained therein. Recipient 106 submits the URL through network 108 to web server 402 (Figure 4) using, for example, a conventional web browser 704 (Figure 7) in recipient 106.

In step 602 (Figure 6), web server 402 (Figure 4) receives the URL and determines that the URL pertains to a secure message to be delivered. As described above, URLs which are not related to secure messages to be delivered by delivery server 104 are redirected to another, conventional HTTP web server or, alternative, are processed by web server 402 in a conventional manner. Web server 402 notifies secure message logic 406 of receipt of the URL.

In step 604 (Figure 6), secure message logic 406 (Figure 4) identifies the secure message identified by the key data within the URL and forms a reference to the secure message.

In step 606, secure message logic 406 retrieves a secure message receiver applet 410 and adds a reference to the secure message to the applet. In particular, secure message logic 406 builds an HTML document which includes a tag to secure message receiver applet 410 and includes the reference as a parameter to be used by secure message receiver applet 410 during execution. In one embodiment, the reference to the secure message is the key data from the URL received in step 602.

In step 608, secure message logic 406 sends the HTML document, with the embedded tag to secure message receiver applet 410, to web server 402 to return to recipient 106 in response to the URL. Secure message receiver applet 410 executes within recipient 106 to handle the secure message in accordance with the secure e-mail protocol in a manner described more completely below.

Recipient 106

Recipient 106 is shown in greater detail in Figure 7. Recipient 106 includes an e-mail reader 702 which is conventional and by which the notification message described above is received and read. In one embodiment, the notification e-mail includes the URL with key data in the form of an HTML attachment as described in U.S. Patent Application S/N 09/386,643 by Jeffrey C. Smith et al. entitled "Electronic Document Delivery System in Which Notification of Said Electronic Documents is Sent to a Recipient Thereof" and that description is incorporated herein by reference. In this illustrative embodiment, the notification message includes the HTML attachment which is opened by recipient 106. Within the computer system of recipient 106, HTML data files such as the HTML attachment are associated with a web browser as the default application with which to open, i.e., process and display the contents of, HTML data files. Thus, in opening the HTML attachment, an HTML browser 704 is caused to execute and display the contents of the HTML attachment. HTML browser 704 is an ordinary web browser in this illustrative embodiment.

In an alternative embodiment, the URL is included as text in the notification message. Some e-mail readers will automatically open an HTML browser in the manner described above when recipient 106 clicks on the URL. In other case, recipient 106 copies the URL from the notification message and independently starts the HTML browser and

enters the URL. It is assumed that recipient 106 is able to use both e-mail reader 702 and HTML browser 704 and to retrieve an HTML document using HTML browser 704 and a URL found in a text message received through e-mail reader 702.

As described above, the HTML attachment or the HTML document sent in response to the URL includes an embedded tag referring to secure message receiver applet 410 (Figure 4) with the reference to the secure e-mail message to be delivered to recipient 106. Embedded tags are known components of HTML data files. In displaying the contents of the HTML attachment, HTML browser 704 (Figure 7) requests secure message receiver applet 410 from web server 402 (Figure 4).

In an alternative embodiment, secure message receiver applet 410 is persistently installed in the computer system of recipient 106. In this alternative embodiment, comparison is made to the version of the persistently installed secure message receiver and the one most recently made available from delivery server 104. Such can be accomplished by initiating execution of the persistently installed secure message receiver which initially requests a newest version number from web server 402 of delivery server 104. If the newest version number is greater than the version of the persistently installed secure message receiver, the persistently installed secure message receiver initiates re-delivery and re-installation of a new secure message receiver.

HTML browser 704 (Figure 7) includes an applet interpreter 706 which, in this illustrative embodiment, is a Java™ interpreter which executes computer instructions according to the Java™ computer instruction language of Sun Microsystems, Inc. Receipt and execution of the computer instructions of the modified applet form secure e-mail receiver 708. Secure e-mail receiver 708 communicates with delivery server 104 to download the secure message and subsequently processes the secure message to render the message readable to recipient 106. Accordingly, secure e-mail receiver 708 requires cryptographic capabilities to decrypt the secure message and to verify the signature of the secure message. In addition, secure e-mail receiver 708 must be able to present the substantive content of the secure message or to save the substantive content to a local disk in such a manner that the substantive content is subsequently accessible, e.g., by storage at a location known recipient 106.

In this illustrative embodiment, secure e-mail receiver 708 is a Java™ Applet that

is cryptographically signed by delivery server 104 and that is transparently and dynamically downloaded via HTML browser 704 by virtue of the recipient simply accessing the URL included in the notification message. Secure e-mail receiver 708 is shown in greater detail in Figure 8.

Secure e-mail receiver 708 includes a communication module 802 which is responsible for communicating with delivery server 104. In this illustrative embodiment, communication module 802 is implemented using the known java.net.URLConnection class that supports the HTTP protocol and that enables retrieval of data from, and sending data to, a web server such as web server 402 (Figure 4). Communication module 802 (Figure 8) retrieves the secure message from web server 402 and, after processing of the secure message as described below, reports the success of such processing to web server 402 in this illustrative embodiment.

Secure e-mail receiver 708 also includes a secure protocol module 804 which is responsible for decrypting the secure message, verifying the sender's signature, and extracting data files contained in the secure message. Secure protocol module 804 also provides supporting PKI functionality including key management, decryption, and signature and certificate verification as described more completely below.

Communication module 802 moves the content of the secure message from communication module 802 to secure protocol module 804 through a pipe 806 in this illustrative embodiment. Communication module 802 streams the secure message through pipe 806 as the secure message is received such that the secure message can be processed simultaneously with receipt of the secure message. Such significantly improves performance of secure e-mail receiver 708. Pipes are known inter-module communications mechanisms and can be implemented using, for example, the known java.io package for classes.

Processing by secure e-mail receiver 708 is shown in logic flow diagram 900. In step 902, secure e-mail receiver 708 initializes communication module 802. In step 904, secure e-mail receiver 708 instructs communication module 802 to retrieve the secure message from delivery server 104. In step 906, secure e-mail receiver 708 forms pipe 806 between communication module 802 and secure protocol module 804 such that data received by communication module 802 are sent immediately to secure protocol module

804. If processing of the received secure message by secure protocol module 804 is successful, secure e-mail receiver 708 reports such success to web server 402 (Figure 4) in step 908.

In this illustrative embodiment, secure e-mail receiver 708 also provides a user interface by which recipient 106 can save to local persistent storage, and subsequently retrieve and open, any data files contained in. and recovered from, the secure message. Furthermore, secure e-mail receiver 708 reports any errors occurring during the processing of the secure message by secure protocol module 804 to recipient 106.

Secure Protocol Module 804

Processing by secure protocol module 804 is shown in logic flow diagram 1000 (Figure 10). As described above with respect to logic flow diagram 300 (Figure 3), logic flow diagram 1000 (Figure 10) represents but one illustrative example of processing according to a secure e-mail protocol. In test step 1002, secure protocol module 804 determines whether the secure message is encrypted. If the secure message is not encrypted, secure protocol module 804 skips steps 1004-1006.

In step 1004, secure protocol module 804 decrypts the secure message. Step 1004 is the inverse of step 314 (Figure 3) and the result is a MIME body. In step 1006 (Figure 10), secure protocol module 804 parses the MIME body. If the secure message is signed, the result of such parsing is the signed data produced in step 308 (Figure 3). Conversely, if the secure message is not signed, the result of such parsing is the collection of data files collected in step 302 (Figure 3).

In test step 1008 (Figure 10), secure protocol module 804 determines whether the secure message is signed. If the secure message is not signed, secure protocol module 804 skips steps 1010-1012.

In step 1010, secure protocol module 804 verifies the signature of the secure message. Step 1010 is the inverse of step 308 (Figure 3). The result of the verification of step 1010 (Figure 10) is the MIME body produced in step 304 (Figure 3). In addition, secure protocol module 804 reports the results of such signature verification to recipient 106, either as a report displayed to a user or as a report communicated to a process receiving the secure message. The report indicates whether the signature is verified and, if

so, the identity of the sender as indicated by the signature. In one embodiment, the report includes the certificate of the sender.

In step 1012 (Figure 10), secure protocol module 804 parses the MIME body produced in step 1010. The result of such parsing is the collection of data files collected in step 302 (Figure 3).

Thus, the result of processing according to logic flow diagram 1000 (Figure 10) by secure protocol module 804 is the collection of data files collected in step 302 (Figure 3) either in step 1012 if the secure message is signed or in step 1006 if the secure message is not signed.

To carry out steps 1004 and 1010, secure protocol module 804 requires some level of public key infrastructure. In particular, secure protocol module 804 requires the ability to manage keys on behalf of recipient 106 and access to certificates from trusted certificate authorities.

To receive secure e-mail according to many secure e-mail protocols such as S/MIME, PGP/MIME, and OpenPGP, recipient 106 must have a private/public key pair. Asymmetric key encryption is known but is described here briefly for completeness. Asymmetric key encryption involves a pair of reciprocal keys; one is kept private and the other is made public and can be widely distributed. The keys are reciprocal in that data encrypted with either key can only be decrypted with the other key.

To encrypt a data file for recipient 106, the data file is encrypted with the public key of recipient 106. The public key of recipient 106 can be obtained from a trusted certificate authority for example. Trusted certificate authorities are known but are described briefly below for completeness. Once the data file is encrypted with the public key of recipient 106, the data file can only be decrypted using the private key of recipient 106. Since the private key of recipient 106 is held in secrecy by recipient 106, only recipient 106 can decrypt the data file. Even the entity encrypting the data file using the public key of recipient 106 cannot reverse the encryption to recover the data file in a cleartext, unencrypted form.

To sign a data file, the data file — or a hash thereof — is encrypted using the private key of the signing entity. As a result, any person or process with access to the public key of the signing entity can decrypt the data file, or the hash thereof. Successful

decryption of the data file or hash using the public key of the signing entity verifies the signature since only the signing entity has access to the private key of the signing entity. Accordingly, the signing entity is authenticated as the source of the data file.

Public keys are frequently stored in the form of a certificate. Certificates are well known but are described briefly for completeness. A certificate combines data identifying an entity, such as a person or computer process, with a public key of the entity. A certificate authority signs the certificate to ensure that the certificate has not been tampered with. Thus, a certificate provides a relatively high degree of confidence in the association between the public key and the identity of the key owner.

Currently available web browsers support secure communications such as SSL (Secure Sockets Layer) and HTTPS (HyperText Transfer Protocol Secure) which use private keys and certificates. Creation of such private keys and certificates is part of the initial configuration of such web browsers.

In this illustrative embodiment, secure protocol module 804 imports such private keys and certificates from HTML browser 704 (Figure 7). Such certificates are communicated to delivery server 104 (Figure 1) and can be used, by sender 102 for example, in encrypting data for recipient 106 in step 314 (Figure 3). In addition, secure protocol module 804 (Figure 8) uses such imported private keys to decrypt such data in step 1004 (Figure 10).

In addition to importing certificates and keys from HTML browser 704 (Figure 7), secure protocol module 804 provides a user interface by which recipient 106 can view and manipulate keys. For example, recipient 106 can locate and add public keys from other entities to a keyring, i.e., a data structure which includes a number of public keys. In addition, the user interface provided by secure protocol module 804 in this illustrative embodiment allows recipient 106 to delete keys from the keyring. The user interface provided by secure protocol module 804 in this illustrative embodiment also allows recipient 106 to export keys to, and import keys from, a password-protected data file stored on persistent storage accessible by recipient 106.

As described briefly above, secure protocol module 804 receives certificates from a trusted certificate authority. The certificate authority provides certificates which can be used by secure protocol module 804 to verify a signature as in step 1010 (Figure 10). A

number of certificate authorities currently exist and have established a reputation for being trustworthy, i.e., for faithfully producing certificates submitted to the certificate authority. In one embodiment, secure protocol module 804 (Figure 8) receives an initial list of trusted certificate authorities and allows recipient 106, by providing a user interface for example, to modify the list to include other certificate authorities trusted by recipient 106. It is important that the initial list of trusted certificate authorities for secure protocol module 804 is secure from tampering by attackers hoping to assume identities of other entities by manipulation of certificates. Accordingly, in this illustrative embodiment, secure e-mail receiver 708 is received from delivery server 104 in a signed form, i.e., as signed by delivery server 104. Therefore, tampering with the contents of secure e-mail receiver 708, including a list of trusted certificate authorities, is prevented.

In an alternative embodiment, a list of trusted certificate authorities can be retrieved from web server 402 (Figure 4) of delivery server 104 each time such a list is needed by secure protocol module 804 (Figure 8). In this alternative embodiment, the list is not modified by recipient 106. In addition, it is important that the list of trusted certificate authorities received from web server 402 is secure from tampering by attackers hoping to assume identities of other entities by manipulation of certificates. Accordingly, in this alternative embodiment, the list of trusted certificate authorities is received from delivery server 104 through a secure channel such as an SSL connection. Therefore, tampering with the list of trusted certificate authorities during delivery of the list is prevented.

In addition to access to keys and certificates, secure protocol module 804 can secure store such keys and certificates within persistent storage of the computer system of recipient 106. In the one embodiment, the private keys are stored locally by recipient 106 using the known PKCS #12 password-based encryption and a strong symmetric encryption algorithm such as the known Triple-DES algorithm.

As described above, the secure e-mail protocol supported by delivery server 104 is the S/MIME protocol in one embodiment. Accordingly, secure protocol module 804 in this embodiment is implemented using the RSA BSAFE Crypto-J™ 2.0 toolkit from RSA Data Security, Inc. This toolkit provides basic cryptographic functionality based on the known PKCS #1 standard. In addition, this toolkit includes modules which can generate

and verify message digests and digital signatures as well as encrypt and decrypt data using a number of encryption algorithms. Familiarity with this toolkit and the S/MIME protocol in general are helpful in implementing this illustrative embodiment of secure protocol module 804.

In addition, the following components are known and are helpful in implementing this illustrative embodiment of secure protocol module 804:

(1) ASN.1: The ASN.1 DER data format provides a universal way to represent and encode structured data. ASN.1 is used widely in public key cryptography. Secure protocol module 804 can parse and generate ASN.1 DER encoding in this illustrative embodiment.

(2) PKCS #1: PKCS #1 can calculate and verify message digests and digital signatures as well as encrypt and decrypt data using a number of encryption algorithms. PKCS #1 is provided by RSA BSAFE Crypto-J™ toolkit.

(3) X.509: X.509 specifies a format for certificates. In this embodiment, PKCS #1 can parse and verify X.509 v3 certificates as well as build a certificate chain. This component of PKCS #1 also provides support for v3 certificate extensions required by S/MIME. X.509 can be implemented using ASN.1 for certificate parsing and PKCS #1 for verification.

(4) PKCS #12: PKCS #12 specifies password-protected data. By implementation of PKCS #12, secure protocol module 804 can extract private keys and certificates from password-protected PKCS # 12 data. In addition, secure protocol module 804 can also generate password-protected PKCS #12 data for a given set of keys and certificates, and a given password. PKCS #12 is implemented based on ASN.1, PKCS #1, and X.509.

(5) Private Key Database: By implementing a private key database, secure protocol module 804 manages the private key or keys of recipient 106. In this illustrative embodiment, secure protocol module 804 supports adding, deleting, enumerating of the private keys. The private keys are stored protected by a user-defined password. While the term "password" can be considered to connote a single word, it is appreciated that "password" as used herein can include non-textual data and can include multiple words such as a passphrase. Secure protocol module 804 uses PKCS #12 to store the keys in the database and to import the keys from other applications.

(6) Trusted Certificate Authority Database: By implementing a trusted certificate

authority database, secure protocol module 804 can access the list of trusted certificate authorities. Secure protocol module 804 uses X.509 to manipulate certificates of the certificate authorities.

(7) PKCS #7 Signed Data: PKCS #7 specifies a cryptographic message syntax. By implementing PKCS #7, secure protocol module 804 can verify the data signed using the PKCS #7 format and can extract the signed content and signer's signature of such data. Secure protocol module 804 uses ASN.1 for parsing of PKCS #7 data; PKCS #1 for verifying the signature; and X.509 and the trusted certificate authority for verifying the signer's certificate.

(8) PKCS #7: PKCS #7 also specifies a format for enveloped data. By implementing the enveloped data portion of PKCS #7, secure protocol module 804 can decrypt data which is encrypted and packaged using the PKCS #7 format. Secure protocol module 804 uses ASN.1 for parsing of PKCS #7 data, the private key database for accessing the private key of recipient 106 needed to decrypt the secure message, and PKCS #1 for decryption.

(9) MIME Parser: Secure protocol module 804 can parse MIME messages and extract any attachments included in such MIME message.

(10) S/MIME Message Processor: Secure protocol module 804 implements a S/MIME message parser which processes S/MIME messages with decryption and signature verification applied as necessary. The S/MIME message parser relies on the MIME parser of secure protocol module 804 to detect and extract S/MIME content as well as to process the original content once all of the cryptographic layers of signing and encryption are removed by using both aspects of PKCS #7 described above as components (7) and (8).

(11) Certificate User Interface: Secure protocol module 804 implements a certificate user interface which can report certificate information to recipient 106. The certificate user interface relies on X.509 to provide such information.

(12) Signature User Interface: Secure protocol module 804 implements a signature user interface which can report S/MIME package signature information to recipient 106. The signature user interface can rely on the certificate user interface for displaying the signer's certificate and on signature info produced by PKCS #7 including, for example, the

signing date and time.

(13) Private Key Database User Interface: Secure protocol module 804 implements a private key database user which provides the user with access to private key database operations. The private key database user interface uses the certificate user interface to show a user's certificate and the private key database to carry out the operations.

The above description is illustrative only and is not limiting. The present invention is limited only by the claims which follow.

What is claimed is:

1. A method for delivering an e-mail message which comports with a secure e-mail protocol to a recipient, the method comprising:

receiving, from the recipient through a computer network, a reference to the e-mail message;

sending, to the recipient through the computer network, computer instructions which, when executed, retrieve the e-mail message and process the message in accordance with the secure e-mail protocol.

2. The method of Claim 1 further comprising:

receiving the e-mail message from a sender through the computer network;

forming the reference to the e-mail message;

including the reference in a notification message; and

sending the notification message to the recipient through the computer network.

3. The method of Claim 1 wherein the computer instructions collectively define an applet.

4. The method of Claim 1 wherein sending comprises sending the computer instructions in such a manner that causes the computer instructions to be executed upon receipt by the recipient without human intervention.

5. The method of Claim 1 wherein the secure e-mail protocol is an S/MIME secure e-mail protocol.

6. The method of Claim 1 wherein the secure e-mail protocol is a PGP/MIME secure e-mail protocol.

7. The method of Claim 1 wherein the secure e-mail protocol is an OpenPGP secure e-mail protocol.

8. The method of Claim 1 wherein receiving is according to a hypertext transfer protocol.

9. The method of Claim 1 wherein sending is according to a hypertext transfer protocol.

10. The method of Claim 1 wherein the e-mail message includes one or more attached data files.

11. The method of Claim 1 wherein the computer instructions, when executed upon receipt by the recipient, process the e-mail message by decrypting at least a portion of the e-mail message.

12. The method of Claim 1 wherein the computer instructions, when executed upon receipt by the recipient, process the e-mail message by verifying a signature of at least a portion of the e-mail message.

13. The method of Claim 1 wherein the computer instructions provide a user interface for management of one or more encryption keys.

14. The method of Claim 1 wherein the computer instructions, when executed, import one or more encryption keys.

15. A computer readable medium useful in association with a computer which includes a processor and a memory, the computer readable medium including computer instructions which are configured to cause the computer to deliver an e-mail message which comports with a secure e-mail protocol to a recipient by:

receiving, from the recipient through a computer network, a reference to the

e-mail message;

sending, to the recipient through the computer network, retrieval computer instructions which, when executed, retrieve the e-mail message and process the message in accordance with the secure e-mail protocol.

16. The computer readable medium of Claim 15 wherein the computer instructions are configured to cause the computer to deliver the e-mail message by also:

receiving the e-mail message from a sender through the computer network;

forming the reference to the e-mail message;

including the reference in a notification message; and

sending the notification message to the recipient through the computer network.

17. The computer readable medium of Claim 15 wherein the retrieval computer instructions collectively define an applet.

18. The computer readable medium of Claim 15 wherein sending comprises sending the retrieval computer instructions in such a manner that causes the retrieval computer instructions to be executed upon receipt by the recipient without human intervention.

19. The computer readable medium of Claim 15 wherein the secure e-mail protocol is an S/MIME secure e-mail protocol.

20. The computer readable medium of Claim 15 wherein the secure e-mail protocol is a PGP/MIME secure e-mail protocol.

21. The computer readable medium of Claim 15 wherein the secure e-mail protocol is an OpenPGP secure e-mail protocol.

22. The computer readable medium of Claim 15 wherein receiving is according

to a hypertext transfer protocol.

23. The computer readable medium of Claim 15 wherein sending is according to a hypertext transfer protocol.

24. The computer readable medium of Claim 15 wherein the e-mail message includes one or more attached data files.

25. The computer readable medium of Claim 15 wherein the retrieval computer instructions, when executed upon receipt by the recipient, process the e-mail message by decrypting at least a portion of the e-mail message.

26. The computer readable medium of Claim 15 wherein the retrieval computer instructions, when executed upon receipt by the recipient, process the e-mail message by verifying a signature of at least a portion of the e-mail message.

27. The computer readable medium of Claim 15 wherein the retrieval computer instructions provide a user interface for management of one or more encryption keys.

28. The computer readable medium of Claim 15 wherein the retrieval computer instructions, when executed, import one or more encryption keys.

29. A computer system comprising:

a processor;

a memory operatively coupled to the processor; and

a delivery module (i) which executes in the processor from the memory and (ii) which, when executed by the processor, causes the computer to deliver an e-mail message which comports with a secure e-mail protocol to a recipient by:

receiving, from the recipient through a computer network, a reference to the e-mail message;

sending, to the recipient through the computer network, retrieval

computer instructions which, when executed, retrieve the e-mail message and process the message in accordance with the secure e-mail protocol.

30. The computer system of Claim 29 wherein the delivery module is configured to cause the computer to deliver the e-mail message by also:

receiving the e-mail message from a sender through the computer network;

forming the reference to the e-mail message;

including the reference in a notification message; and

sending the notification message to the recipient through the computer network.

31. The computer system of Claim 29 wherein the retrieval computer instructions collectively define an applet.

32. The computer system of Claim 29 wherein sending comprises sending the retrieval computer instructions in such a manner that causes the retrieval computer instructions to be executed upon receipt by the recipient without human intervention.

33. The computer system of Claim 29 wherein the secure e-mail protocol is an S/MIME secure e-mail protocol.

34. The computer system of Claim 29 wherein the secure e-mail protocol is a PGP/MIME secure e-mail protocol.

35. The computer system of Claim 29 wherein the secure e-mail protocol is an OpenPGP secure e-mail protocol.

36. The computer system of Claim 29 wherein receiving is according to a hypertext transfer protocol.

37. The computer system of Claim 29 wherein sending is according to a

hypertext transfer protocol.

38. The computer system of Claim 29 wherein the e-mail message includes one or more attached data files.

39. The computer system of Claim 29 wherein the retrieval computer instructions, when executed upon receipt by the recipient, process the e-mail message by decrypting at least a portion of the e-mail message.

40. The computer system of Claim 29 wherein the retrieval computer instructions, when executed upon receipt by the recipient, process the e-mail message by verifying a signature of at least a portion of the e-mail message.

41. The computer system of Claim 29 wherein the retrieval computer instructions provide a user interface for management of one or more encryption keys.

42. The computer system of Claim 29 wherein the retrieval computer instructions, when executed, import one or more encryption keys.

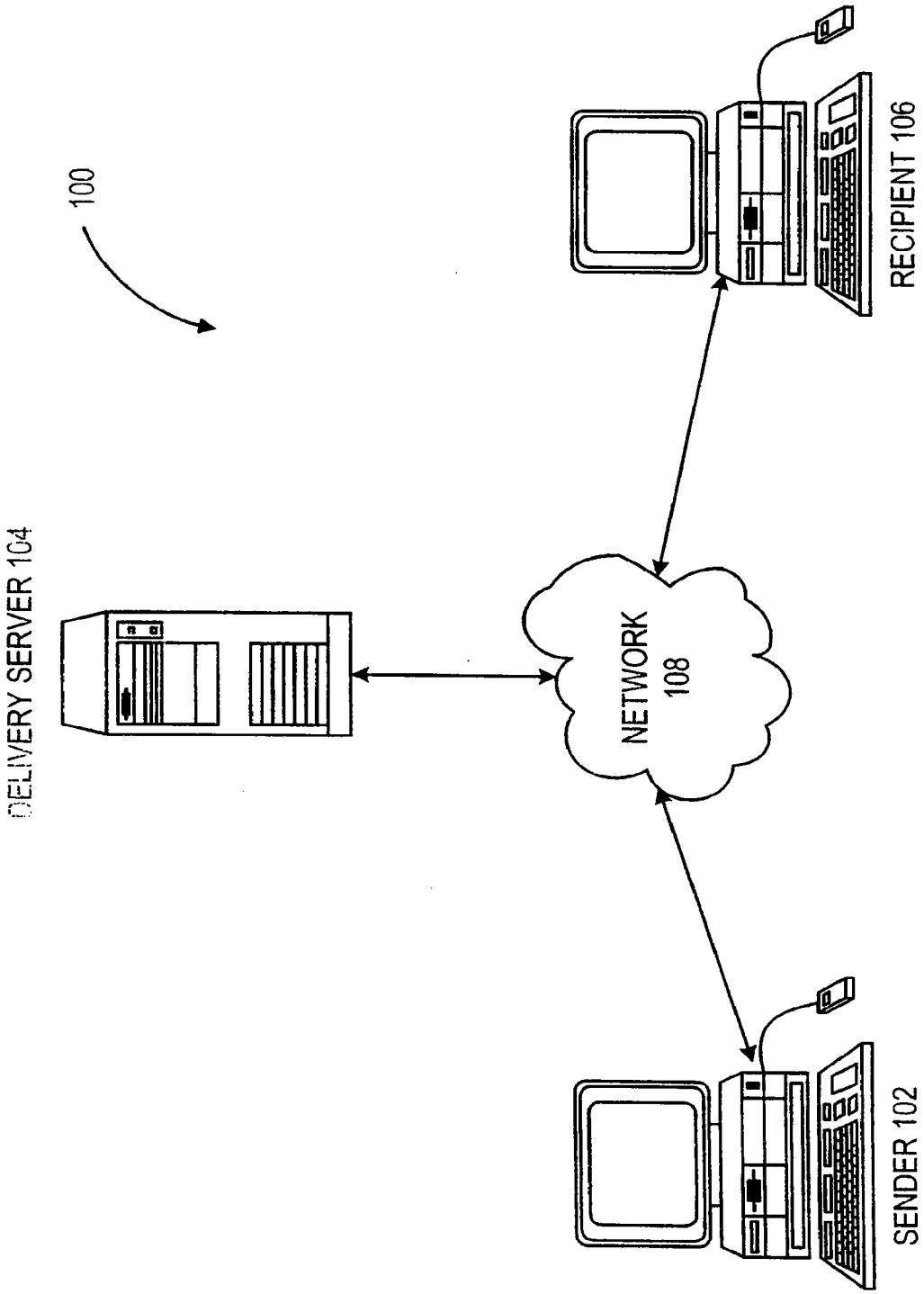


FIGURE 1

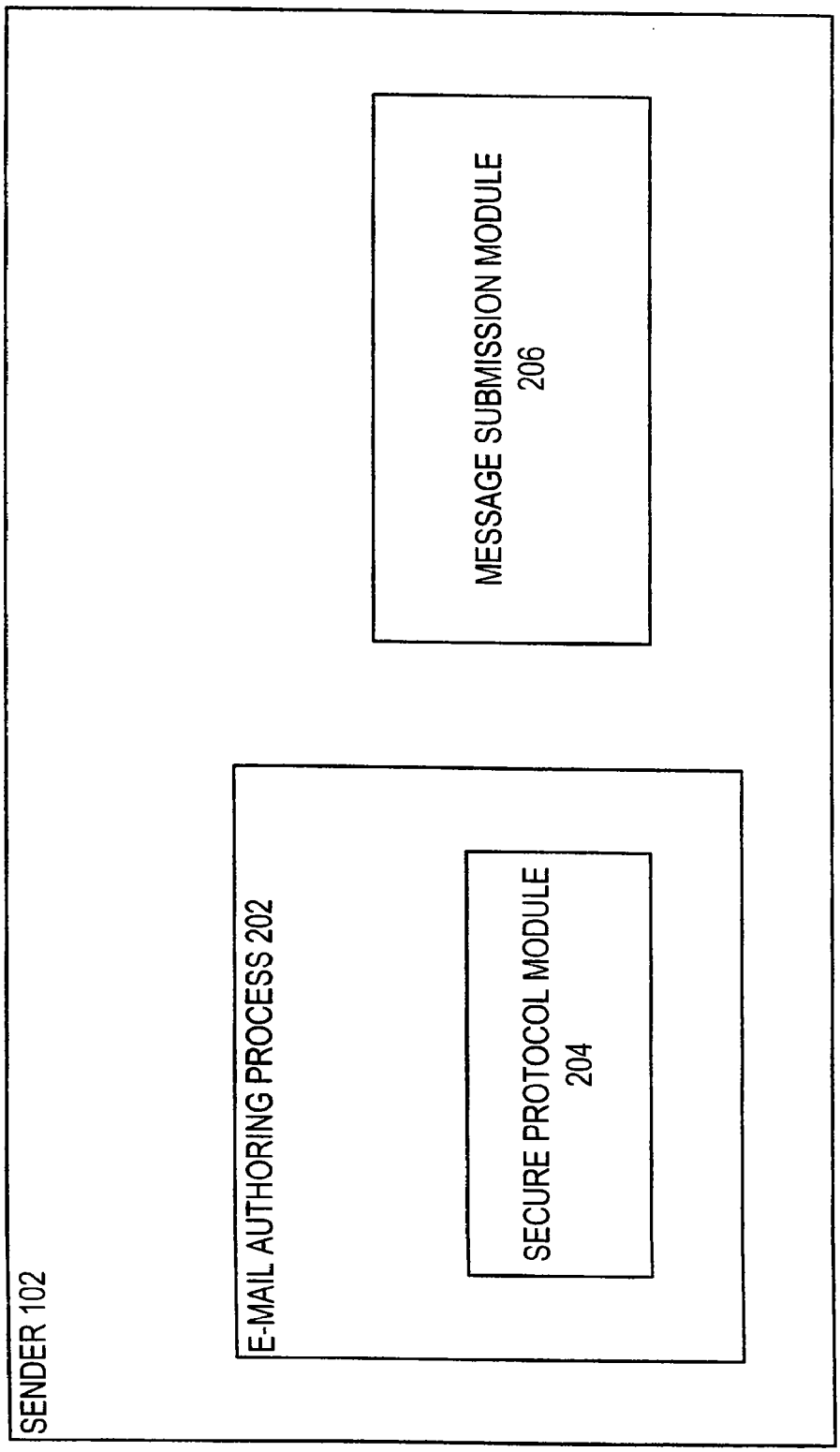


FIGURE 2

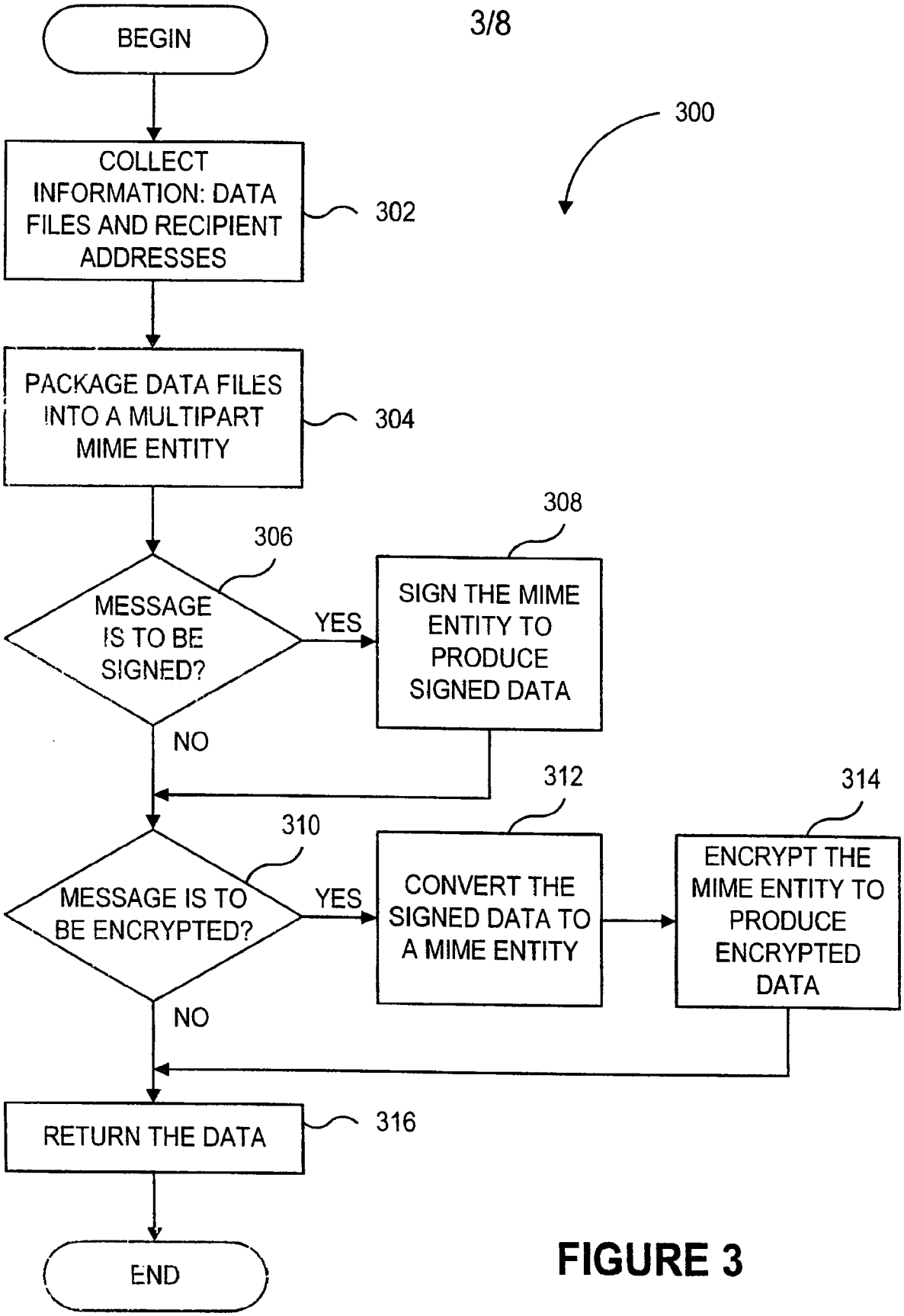


FIGURE 3

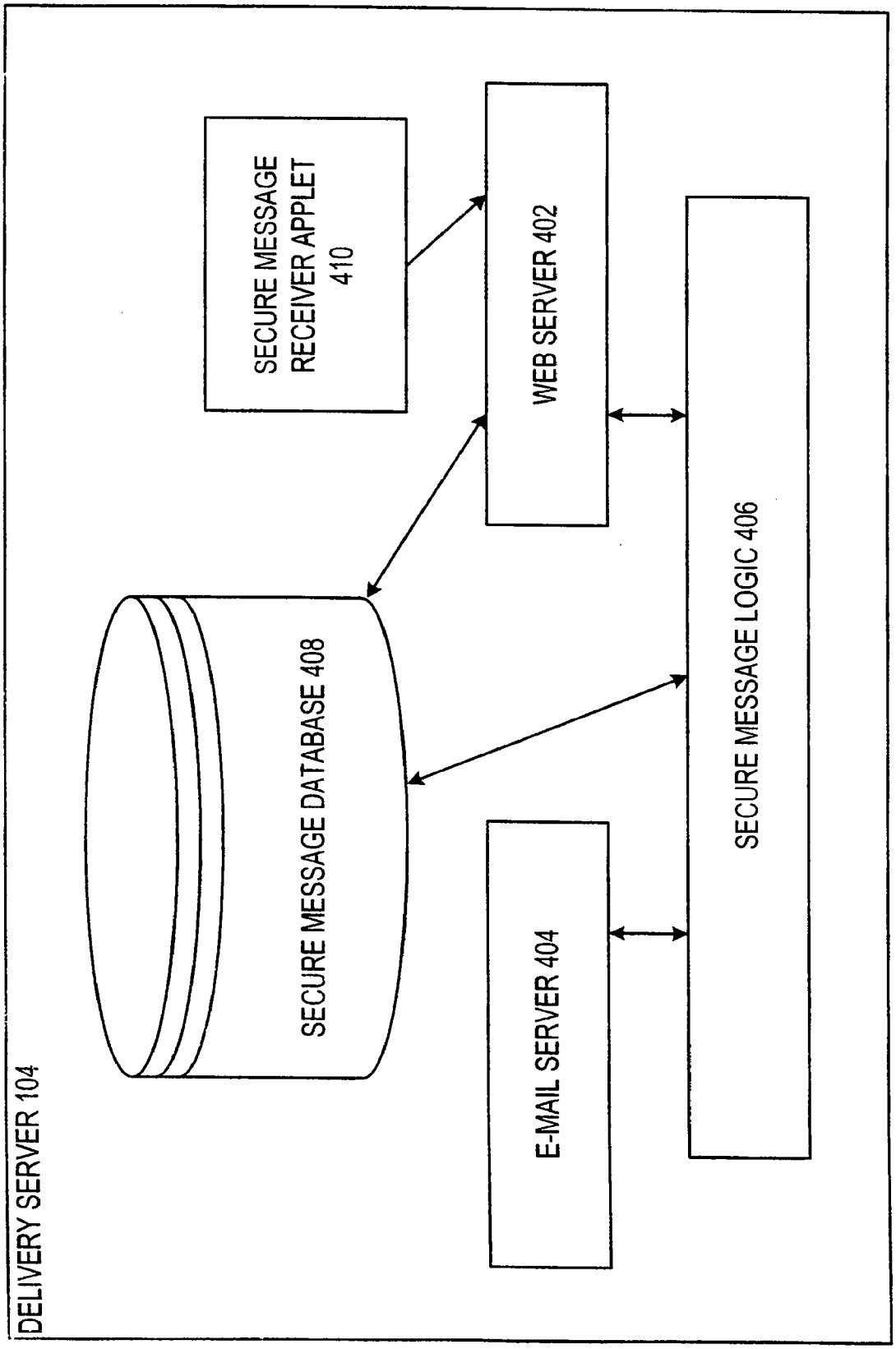


FIGURE 4

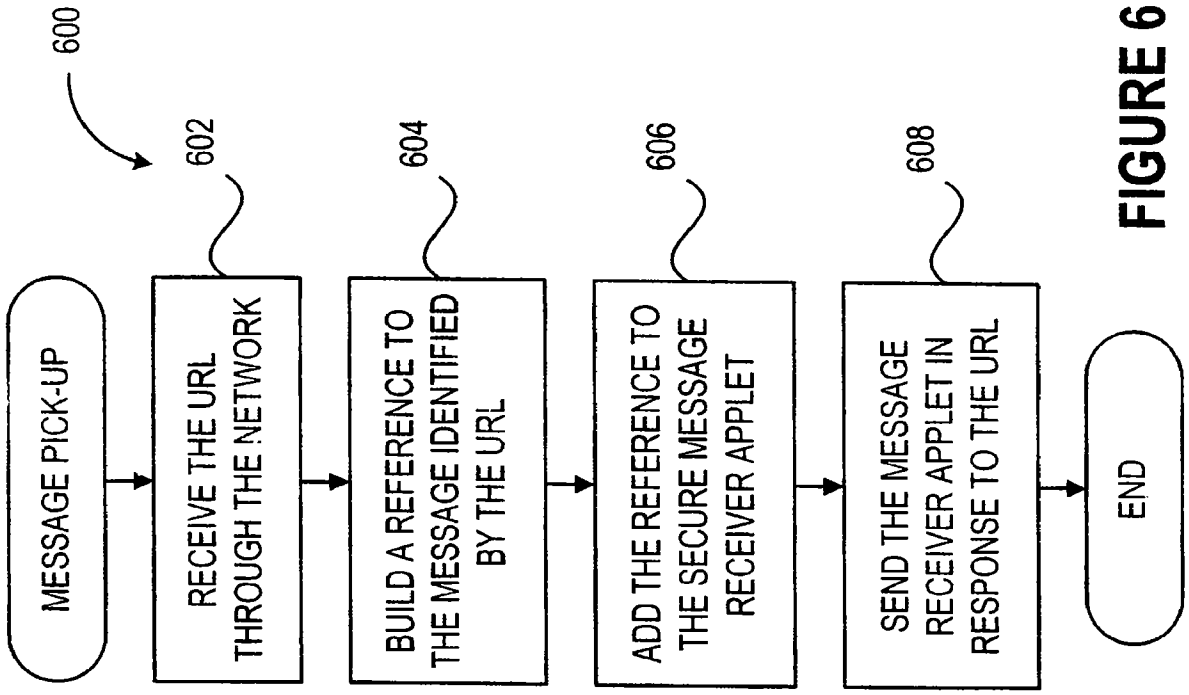


FIGURE 6

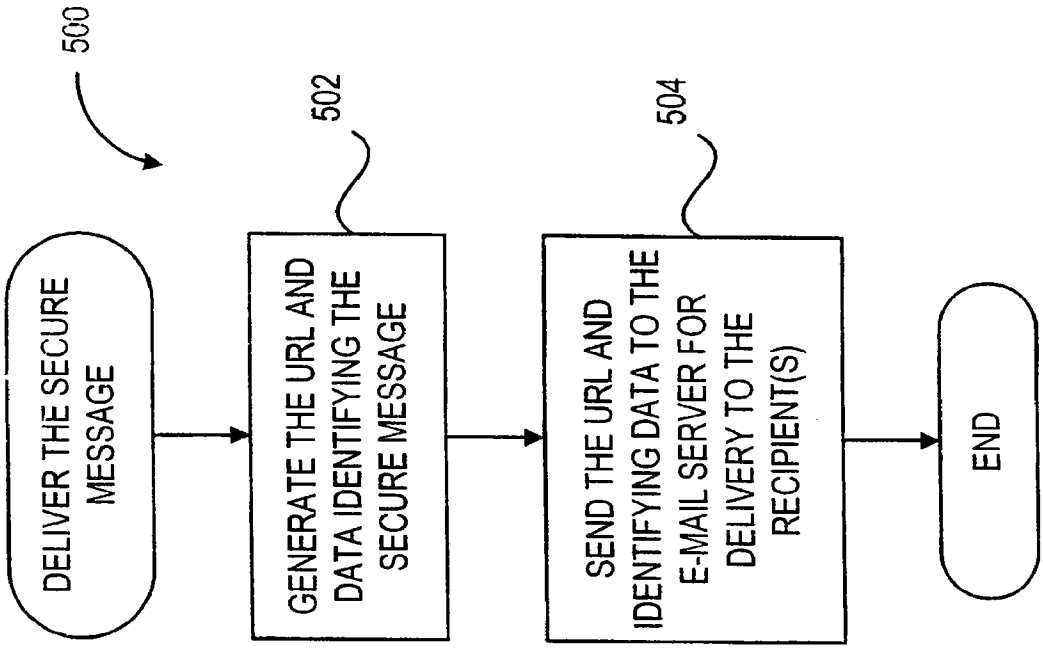


FIGURE 5

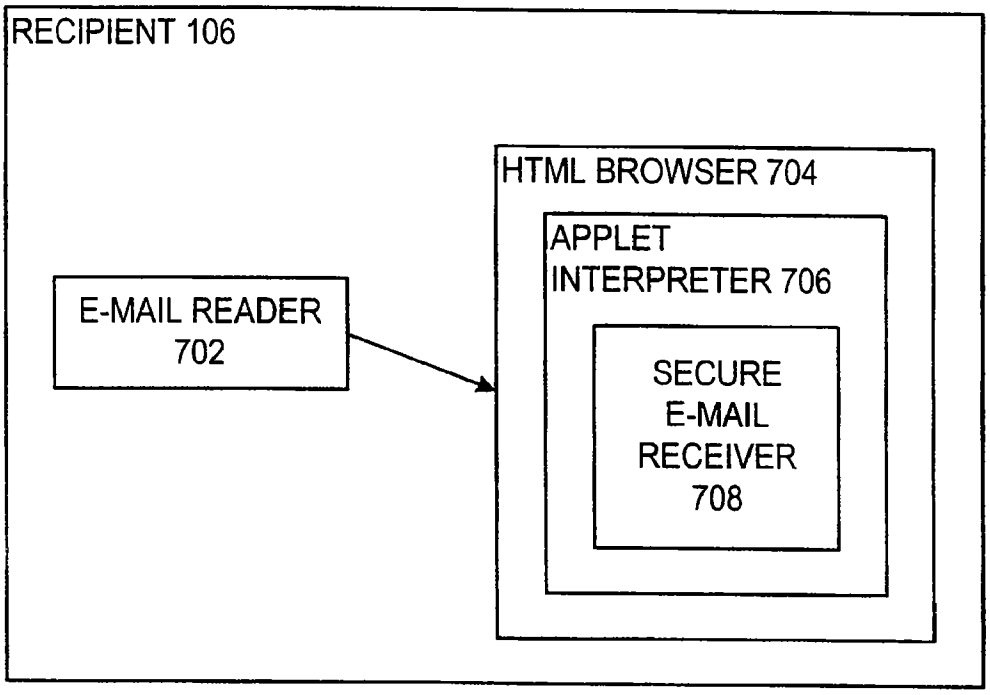


FIGURE 7

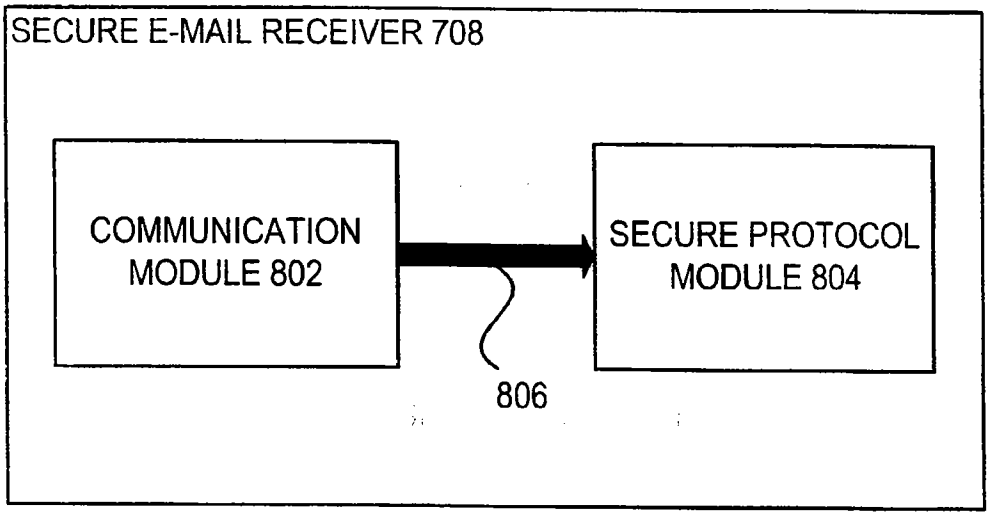


FIGURE 8

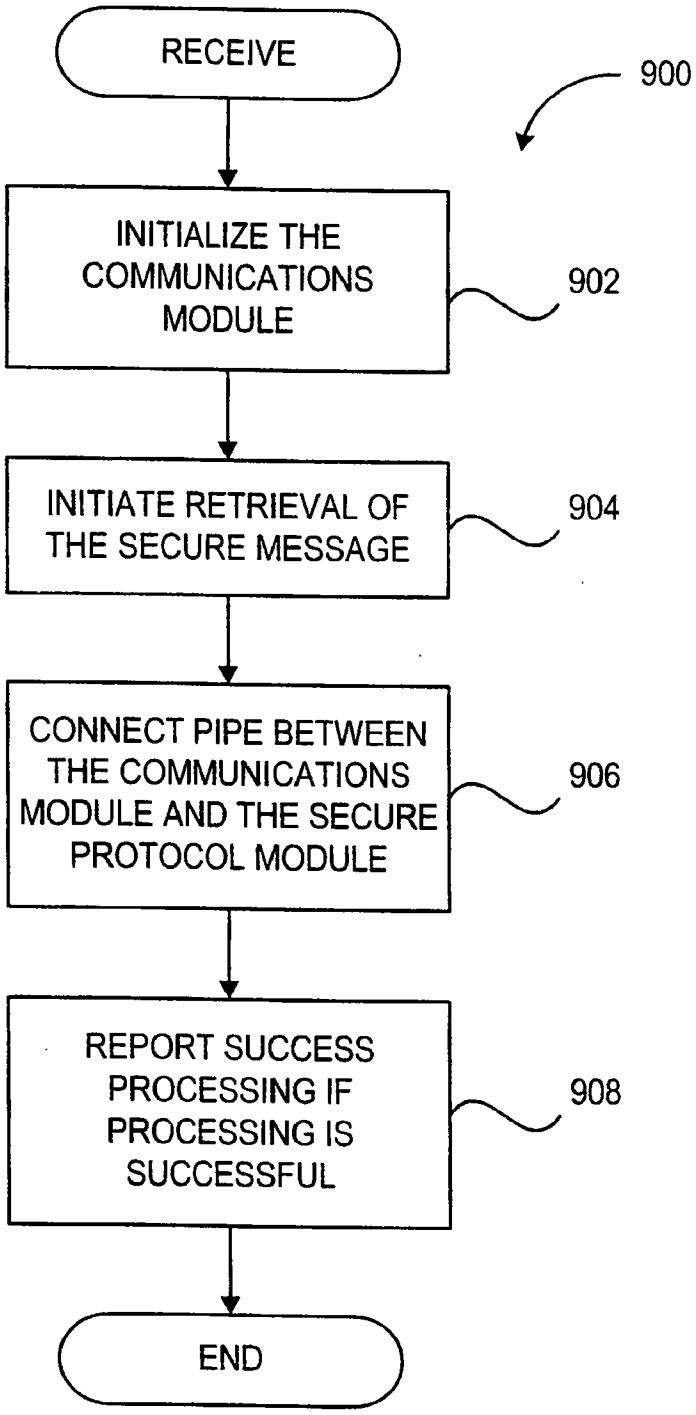


FIGURE 9

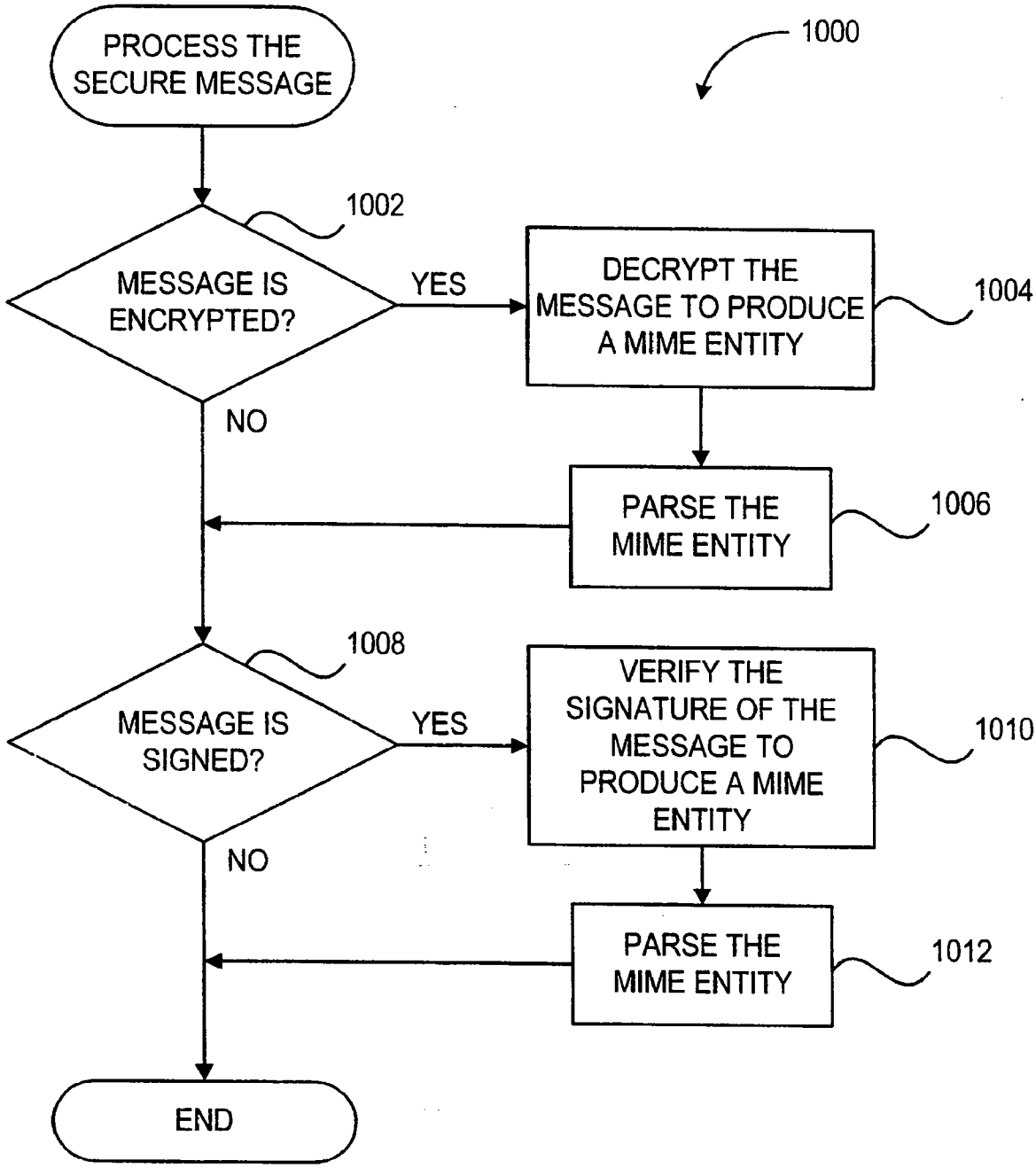


FIGURE 10

INTERNATIONAL SEARCH REPORT

	International Application No PCT/US 00/01011

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L12/58 H04L9/30 H04L9/08 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 869 652 A (TUMBLEWEED SOFTWARE CORP) 7 October 1998 (1998-10-07) page 5, line 29 - line 38 page 12, line 14 -page 13, line 24 page 14, line 52 -page 17, line 30 figure 3 ---	1-42
Y	LEVIEN R: "PROTECTING INTERNET E-MAIL FROM PRYING EYES" DATA COMMUNICATIONS,US,MCGRAW HILL. NEW YORK, vol. 25, no. 6, 1 May 1996 (1996-05-01), pages 117-118,120,122, XP000587586 ISSN: 0363-6399 the whole document --- -/--	1-42

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
19 May 2000	26/05/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Carnerero Álvaro, F

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 00/01011

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	NADEAU M: "YOUR E-MAIL IS OBSOLETE" BYTE,US,MCGRAW-HILL INC. ST PETERBOROUGH, vol. 22, no. 2, 1 February 1997 (1997-02-01), pages 66-69,72,74,, XP000680521 ISSN: 0360-5280 page 66 -page 67 page 69 -page 72 page 78 ---	1,15,29
A	STALLINGS W: "S/MIME: E-MAIL GETS SECURE" BYTE,US,MCGRAW-HILL INC. ST PETERBOROUGH, vol. 23, no. 7, 1 July 1998 (1998-07-01), pages 41-42, XP000774260 ISSN: 0360-5280 the whole document -----	1-42

1

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Ir. ...onal Application No
PCT/US 00/01011

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0869652 A	07-10-1998	JP 11031127 A	02-02-1999

Form PCT/ISA/210 (patent family annex) (July 1992)